REMARKS

Reconsideration of this application, in view of the foregoing amendments and the following remarks, is respectfully requested.

Claims 1-15 were pending for consideration in this application. By the foregoing amendment, Applicant has amended Claims 1, 5, 7, and 12 and added new claim 16. Claims 1-16 are now pending.

Claims 13 and 4-15 were rejected under 35 USC 102(e) as being anticipated by US Patent 6,859,891 to Edwards.

Claim 4 was objected to but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Applicant thanks the Examiner for indicating allowable subject matter and has added new Claim 16 that consists of claim 4 rewritten in independent form including all of the limitations of base claim 1, there were no intervening claims.

Applicant's invention provides a way to reduce the amount of information that is traced during a debugging session. As described in the Summary of Invention, the trigger unit can be programmed to transmit selectively trace streams that are state dependant. Furthermore, one or more of several trace units may be selectively disabled while others are tracing. One application of this capability is described at page 26, beginning at line 25, in which optional tracing of interrupt service routines is described. This is important because often the interrupts may occur somewhat randomly and often are not of interest when a particular portion of instruction code is being debugged. Therefore it is preferable to not trace the execution of the interrupt service routine but to resume tracing when the interrupt service routine is completed. At page 23, beginning at line 20, the specification explains that execution of the interrupt service routine code is also referred to as secondary code execution.

Claim 1 recites: "...timing trace apparatus, the timing trace apparatus responsive to the control signals for selectively providing timing trace streams during secondary code execution..." Applicant has now amended Claim 1 to make it clear that this occurs "while continuing to provide timing trace streams during primary code execution." Applicant finds only one reference to interrupts in Edwards:

> Branch types may include, a conditional branch, an unconditional branch, a return from exception (RTE) branch, and other types of branches including trap launch branches, interrupt launch branches, etc. as known in the art. Branch type 204 is useful when determining the location in software of an error, or providing branching information for coverage analysis tools, call graph profilers, trace analysis tools, or compiler feedback tools. (Col 7, lines7-14)

Thus, Edwards only teaches that tracing continues after an interrupt. Conversely, Applicant's invention is directed to selectively not tracing an interrupt and the branch that occurs due to the interrupt. Edwards clearly does not teach or suggest the novel selective tracing of Claim 1.

Independent Claims 7 and 12 have also been amended to more clearly recite the selective nature of tracing as the processor executes in different states. As discussed above, Edwards does not teach or suggest selective tracing according to a current state of execution.

Claims 2-6, 8-11 and 13-15 depend on allowable base Claims and are allowable for that reason and by virtue of their further distinctive recitations. For example, Claim 5 recites: "the target processor has three states, ... wherein the trigger unit is responsive to the three states to selectively enable and disable the timing trace apparatus and the program counter and data trace apparatus in accordance with a current state of processor execution." Such selective tracing is not suggested by Edwards.

Claim 6 recites "wherein the timing trace stream can be controllably enabled during an execution halt state." As discussed with reference to Applicant's Figures 8A-D, under some conditions it is useful to trace when the processor is placed into a execution halt state. The Examiner points to Edwards at Col 8, line 44. However, here Edwards teaches:

> Stall signal 211 indicates that the processor 102 should stall its execution pipeline. Stalled signal 212 indicates to the debug circuit 103 that the pipeline has stalled. These indications 211, 212 are useful in performing flow control between processor 102 and debug circuit 103. For example, processor 103 may be generating trace information beyond storage capability of debug circuit 103 and indicates to the processor 102 to stall its execution pipeline such that additional trace information cannot be generated. Debug circuit 103 may, for example, stall processor 102 when a trace buffer is within a certain number of entries before it is filled. Thus, debug circuit 103 can avoid losing trace information. Stall signal 211 and stalled signal 212 may each be one bit values.

Thus, Edwards teaches that tracing is stopped during a stall; Edwards does not suggest continued tracing during the stall. Claim 6 and similarly Claim 10 are therefore allowable over Edwards for this additional reason.

Applicant believes this application and the claims herein to be in a condition for allowance and respectfully requests that the Examiner allow this application to pass to the issue branch.

Should the Examiner have further inquiry concerning these matters, please contact the below named attorney for Applicant.

Respectfully submitted,

/Gerald E. Laws/

Gerald E. Laws
Attorney for Applicant
Reg. No. 39268
713-937-8823

Texas Instruments Incorporated
P.O. Box 655474, MS 3999
Dallas, TX 75265
(972) 917-5287